



Algorithm: *StochPrice*

( $P_{init}$ : Initial price,  $T$ : Unit sales period)

1. Initialization
  - 1.1. Initial price:  $p := p_{init}$
  - 1.2. Initial time:  $t := \text{current time}$
  - 1.3. Initial setting of trial number:  $I := 1$
2. Repeat For  $I = 1$  until forever
  - 2.1. Set  $\Delta$  as follows:  $\Delta := I^{-1/3}$
  - 2.2. For a period of  $T$ , set price to  $p + \Delta$
  - 2.3. Let  $S(p+\Delta)$  be the amount of sales during this time.
  - 2.4. For a period of  $T$ , set price to  $p - \Delta$
  - 2.5. Let  $S(p-\Delta)$  be the amount of sales during this time.
  - 2.6. Calculate the obtained profit as follows:  
$$P(p+\Delta) = S(p+\Delta) \cdot (p+\Delta)$$
$$P(p-\Delta) = S(p-\Delta) \cdot (p-\Delta)$$
  - 2.7. Set the update interval  $A$  as follows:  
$$A := \frac{1}{I}$$
  - 2.8. Update the current price as follows:  
$$p := p + \frac{A}{\Delta} \frac{P(p+\Delta) - P(p-\Delta)}{2T}$$
  - 2.9. If necessary, clamp the value of  $p$  between the maximum and minimum possible prices.  
$$p := \min\{p_{max}-\Delta, \max\{p_{min}+\Delta, p\}\}$$
  - 2.10. Update current time  $t$ .  
$$t := t + 2T$$
  - 2.11. Store current time  $t$ , price  $p$  and sales  
$$S(p) = S(p-\Delta) + S(p+\Delta)$$
 to the database.

FIG. 2

Algorithm: FeaturePrice(W<sub>init</sub> : Initial weight vector, T : unit sales period)

1. Initialization
  - 1.1. Initial weight vector:  $W := W_{init}$
  - 1.2. Initial time:  $t :=$  current time
2. Repeat For  $I = 1$  until forever
  - 2.1. For item  $i = 1$  until  $N$  (number of item)
    - 2.1.1.  $X(i) :=$  attribute vector for item  $i$
    - 2.1.2.  $P(i) := W \cdot X(i)$
    - 2.1.3.  $\Delta := I^{-1/3}$
    - 2.1.4.  $\bar{V}(i) = \text{Random-vector}()$
    - 2.1.5.  $\bar{\Delta}(i) = \Delta \cdot (\bar{V}(i) / |\bar{V}(i)|)$
    - 2.1.6. Set current price for item  $i$  as follows:  
 $p(i) := (W + \bar{\Delta}(i)) \cdot X(i)$
    - 2.1.7. For each item  $i$ , If  $p(i)_{min} > (W + \bar{\Delta}(i)) \cdot X(i)$  or  $p(i)_{max} < (W + \bar{\Delta}(i)) \cdot X(i)$  then select maximum positive constant  $\Pi$  which satisfies the following equation and put  $\bar{\Delta}(i) := \Pi \bar{\Delta}(i)$ :  
 $p(i)_{min} \leq (W + \Pi \bar{\Delta}(i)) \cdot X(i) \leq p(i)_{max}$
  - 2.2. For a time period of  $T$ , set the price of each item ( $i$ ) to  $p(i) := (W + \bar{\Delta}(i)) \cdot X(i)$  and conduct sales.
  - 2.3. Let  $S(W + \bar{\Delta}(i))$  be amount of sales thus obtained for each item ( $i$ ).
  - 2.4. For a time period of  $T$ , set the price of each item ( $i$ ) to  $p(i) := (W - \bar{\Delta}(i)) \cdot X(i)$  and conduct sales.
  - 2.3. Let  $S(W - \bar{\Delta}(i))$  be amount of sales thus obtained for each item ( $i$ ).
  - 2.6. For each item  $i$ , calculate total profits based on the above amount of sales.  
 $P(W + \bar{\Delta}(i)) = S(W + \bar{\Delta}(i)) \cdot X(i) (W + \bar{\Delta}(i))$   
 $P(W - \bar{\Delta}(i)) = S(W - \bar{\Delta}(i)) \cdot X(i) (W - \bar{\Delta}(i))$
  - 2.7. For  $i = 1$  until number of items  
 Update the weight vector  $W$  as follows:  

$$W := W + \frac{\Delta}{|\bar{\Delta}(i)|} \frac{P(W + \bar{\Delta}(i)) - P(W - \bar{\Delta}(i))}{2T}$$
  - 2.8. Update current time  $t$ .  
 $t := t + 2T$
  - 2.9. Store current time  $t$ , price  $p$  and  
 $S(p) = S(W - \bar{\Delta}(i)) + S(W + \bar{\Delta}(i))$  to the database.

Algorithm: VarietySelection

( $W$  : weight vector,  $G$  : set of items,  $n$  : number of items to be displayed,  $N$  : Number of iterations)

1. Initialization
  - 1.1. Sort  $G$  in increasing order of  $PTotal(i, W)$
  - 1.2.  $S := \text{First} - n(G, n)$
  - 1.3.  $\bar{S} := G \setminus S$
2. Repeat for  $i = 1$  until  $N$ 
  - 2.1. Randomly select item  $j \in S$ .
  - 2.2. If there exists item  $k$  such that exchanging  $j, k$  would result in increasing the evaluation value of
$$\sum_{i \in S} \lambda_1 PTotal(i, W) + \lambda_2 H(S)$$
then make that exchange and update  $S$  and  $\bar{S}$ .
3. Output  $S$ .

FIG. 4